Remarks

Applicant respectfully requests that this Amendment After Final Action be admitted under 37 C.F.R. § 1.116.

Applicant submits that this response presents claims in better form for consideration on appeal. Furthermore, applicant believes that consideration of this Amendment could lead to favorable action that would remove one or more issues for appeal.

No claims have been amended. No claims have been canceled. Therefore, claims 1-6 and 10-17 are now presented for examination.

Claims 1, 2, 10, 13 and 14 stand rejected under 35 U.S.C. §103(a) as being unpatentable over "How Debuggers Work," by Jonathan B. Rosenberg, 1996, in view of Wallace et al. (U.S. Patent No. 6,018,799) and further in view of Lo et al. (U.S. Patent No. 6,151,706). Applicant submits that the present claims are patentable over any combination of Rosenberg, Wallace and Lo.

Rosenberg discloses stack unwinding, which employs an algorithm for finding traces on a stack. Commands are implemented to unwind a stack to find a parent procedure's frame pointer and return address. See Rosenberg at page 136, lines 23-30. The algorithm for unwinding a traditional stack involves pushing return addresses onto the stack. A procedure call pushes a return address onto the stack and a child procedure pushes the parent's frame pointer address onto the stack page 137, lines 23-29.

Lo discloses a method a system and method for extending sparse partial redundancy elimination (PRE) to support speculative code motion within an optimizing compiler. See Lo at col. 3, ll. 23-25.

Nevertheless, neither Rosenberg nor Lo disclose or suggest determining a set of one or more operations to be inserted into the program in order to set each component of the state at each selected program point, wherein the operations assure that the data structure will be in an accurate state at the selected program points.

In fact, the Examiner has admitted in the Office Action that neither Rosenberg nor Lo disclose or suggest such a feature. See Final Office Action at page 2, paragraph 2. However, the Examiner maintains that Wallace discloses that instructions are inserted into a program in order to set each component of the state at each selected program point. Id.

Wallace discloses enabling a compiler to generate efficient code to access stack registers on a register stack. See Wallace at Abstract. In particular, a pseudo-register mapping process for mapping pseudo-registers to stack registers within the register stack is disclosed. The process includes an `iterate each instruction` procedure, which iterates each instruction in a basic block. Each instruction iterated by the `iterate each instruction` procedure is checked to determine whether the instruction accesses a pseudo-register at a `floating point register instruction` decision procedure. If the iterated instruction accesses a pseudo-register, the process continues to a `stack state change` decision procedure that determines whether the iterated instruction requires a permutation to the register stack (and the associated register stack state). If the `stack state change` decision procedure determines that the iterated instruction requires a stack permutation, the process continues to an `insert register stack permutation instruction` procedure, which inserts instructions to place the register stack in a condition appropriate for the iterated instruction. Next, the register stack state is updated by an `update mapping state` procedure. Thus, the register stack state is responsive to changes in the position of the stack registers in the register stack. Thus, the compiler maintains the state of the register stack responsive to the operation of each instruction that accesses the register stack. See Wallace at col. 12, ll. 25 – col. 13, ll. 17.

Claim 1 of the present application recites:

> For a computer-executable program that operates on a data structure, where the data structure must have a required state at selected program points, a method of transforming said program comprising:

(A) analyzing the program to determine the state of said data structure at said selected program points;

(B) partitioning said determined state at each said program point into components that may each be set separately;

(C) <u>determining operations to be inserted into the program in order to set each component of the state at each selected program point, wherein the operations assure that the data structure will be in an accurate state at the selected program points</u>; and

(D) placing said operations to eliminates partial redundancies of said operations.

As discussed above Wallace does not disclose or suggest determining operations to be inserted into the program in order to set each component of the state at each selected program point, wherein the operations assure that the data structure will be in an accurate state at the selected program points. Particularly, Wallace does not disclose a process of determining operations to be inserted into a program in order to set each component of a state at each selected program point. Instead, Wallace discloses inserting instructions to place a register stack in a condition appropriate for an iterated instruction. <u>Applicant submits that placing a register stack in a condition appropriate for an iterated instruction is not equivalent to inserting operations into a program to set each component of a state at each selected program point</u>.

Since neither Rosenberg, Wallace nor Lo disclose or suggest determining operations to be inserted into the program in order to set each component of the state at each selected program point, wherein the operations assure that the data structure will be in an accurate state at the selected program points, any combination of Rosenberg, Wallace and Lo would also not disclose or suggest such a feature. Thus, claim 1 is patentable over Rosenberg in view of Wallace and further in view of Lo.

Claims 2-6 depend from claim 1 and include additional features. Thus, claims 2-6 are also patentable over Rosenberg in view of Wallace and further in view of Lo.

Claim 10 recites:

For a computer-executable program that operates on a data structure, where the data structure must have a required state at selected program points, a method of transforming said program comprising:

(A) analyzing the program to determine the state of an instance of said data structure at said selected program points;

(B) partitioning said instance of said data structure into components;

(C) <u>determining a set of one or more operations to be inserted into the program in order to set each component of the state at each selected program point, wherein the operations assure that the data structure will be in an accurate state at the selected program points;</u>

(D) computing placement of the set of operations to eliminate partial redundancies; and

(E) inserting the set of operations at said program points according to the computed placement.

Accordingly, for the reasons described above with respect to claim 1, claim 10 is also patentable over Rosenberg in view of Wallace and further in view of Lo. Since claims 11 and 12 depend from claim 11 and include additional features, claims 11 and 12 are also patentable over Rosenberg in view of Wallace and further in view of Lo.

Claim 13 recites:

A machine-readable medium having a set of instructions, which when executed by a set of one or more processors, causes said set of processors to perform operations comprising:

(A) analyzing a program that operates on a data structure, which must have a required state at selected program points in the program, to determine the state of an instance of said data structure at said selected program points;

(B) partitioning said instance of said data structure into components;

(C) determining a set of one or more operations to be inserted into the program in order to set each component of the state at each selected program point, wherein the operations assure that the data structure will be in an accurate state at the selected program points;

(D) computing placement of the set of operations to eliminate partial redundancies; and

(E) inserting the set of operations at said program points according to the computed placement.

Therefore, for the reasons described above with respect to claim 1, claim 13 is also patentable over Rosenberg in view of Wallace and further in view of Lo. Since claims 14-17 depend from claim 13 and include additional limitations, claims 14-17 are also patentable over Rosenberg in view of Wallace and further in view of Lo.

Claims 3 and 15 stand rejected under 35 U.S.C. §103(a) as being unpatentable over Rosenberg in view of Wallace and further in view of Lo and Gordon et al. (U.S. Patent No. 6,507,805). Applicant submits that the present claims are patentable over any combination of Rosenberg, Wallace and Lo even in view of Gordon.

Gordon discloses a method for building a call stack tree for in a software program. See Gordon at col. 18, ll. 25-30. Nonetheless, Gordon does not disclose or suggest determining a set of one or more operations to be inserted into the program in order to set each component of the state at each selected program point, wherein the operations assure that the data structure will be in an accurate state at the selected program points.

As discussed above, neither Rosenberg, Wallace nor Lo disclose or suggest determining a set of one or more operations to be inserted into the program in order to set each component of the state at each selected program point, wherein the operations assure that the data structure will be in an accurate state at the selected program points. As a result, any combination of Rosenberg, Wallace, Lo and Gordon would also not disclose or suggest such a limitation. Thus, the present claims are patentable over any combination of Rosenberg, Wallace, Lo and Gordon would also not disclose or suggest such a limitation.

Claims 4, 5, 11, 12, 16 and 17 stand rejected under 35 U.S.C. §103(a) as being unpatentable over Rosenberg in view of Lo and further in view of Dunn et al. (U.S.

Patent No. 6,247,172). Applicant submits that the present claims are patentable over Rosenberg and Lo even in view of Dunn.

Dunn discloses a translating software emulator designed for converting code from a legacy system to a target system and fully preserving the synchronous exception state while allowing for full and aggressive optimization in the translation. See Dunn at Abstract. However, Dunn does not disclose or suggest determining a set of one or more operations to be inserted into the program in order to set each component of the state at each selected program point, wherein the operations assure that the data structure will be in an accurate state at the selected program points.

As discussed above, neither Rosenberg nor Lo disclose or suggest such a limitation. Therefore, any combination of Rosenberg, Lo and Dunn would also not disclose or suggest determining a set of one or more operations to be inserted into the program in order to set each component of the state at each selected program point, wherein the operations assure that the data structure will be in an accurate state at the selected program points.

Claims 6 and 18 stand rejected under 35 U.S.C. §103(a) as being unpatentable over Rosenberg in view of Lo and further in view of Dunn and Gordon. Applicant submits that the present claims are patentable over any combination of Rosenberg, Lo, Dunn and Gordon as described above since none of these references disclose or suggest determining a set of one or more operations to be inserted into the program in order to set each component of the state at each selected program point, wherein the operations assure that the data structure will be in an accurate state at the selected program points.

Applicant respectfully submits that the rejections have been overcome, and that the claims are in condition for allowance. Accordingly, applicant respectfully requests the rejections be withdrawn and the claims be allowed.

The Examiner is requested to call the undersigned at (303) 740-1980 if there remains any issue with allowance of the case.

Please charge any shortage to our Deposit Account No. 02-2666.

Respectfully submitted,
BLAKELY, SOKOLOFF, TAYLOR & ZAFMAN LLP

[illegible]

Date: March 30, 2004

Mark L. Watson
Reg. No. 46,322

12400 Wilshire Boulevard
7th Floor
Los Angeles, California 90025-1026
(303) 740-1980